Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom

TELEPHONE: **Cambridge (01223) 515010**
INTERNATIONAL: **+44 1223 515010**
FAX: **+44 1223 359779**
E-MAIL: **apm@ansa.co.uk**

ANSA Phase III

Real Time ANSAware: ANSAworks'95 presentation

Rob van der Linden

Abstract

Distributed computing systems are increasingly used as enablers for new business opportunities. New products and services are emerging in local and wide area networks such as the information superhighway. There is significant pressure to combine voice and video with traditional data into new distributed interactive multi-media applications

Personal workstations already support multi-media applications in a local context. Advances in network technology allow the simultaneous transfer of voice, video and data. To provide any real time guarantees in an open distributed system, it must be possible to protect local resources from unknown demands emerging from other potentially remote applications and from the network.

ANSAware/RT 1.0 is the result of the first set of experiments in combining real time processing techniques within ANSA's architecture for open distributed processing. It offers low level resource separation with quality of service (QoS) control. An extended programming model allows high level program control over resource allocation. ANSAware/RT 1.0 is available for DEC OSF/1 (versions 1.3 and 3.0) and offers significant performance improvements over ANSAware 4.1. A version for Windows NT is under development.

APM.1452.01 **Approved** 31st March 1995
External Paper

Distribution:
Supersedes:
Superseded by:

Real Time ANSAware

ANSAware/RT 1.0

(/RT features designed by Guangxing Li)

presented by:
Rob van der Linden

ANSAworks ‘95

3 - 5 April 1995

Overview

- Introduction
- Engineering and Programming models
- Performance
- Availability

Market drivers

- **Distributed computing: enabler for new business**
  - **require service delivery guarantees**
- **Increasing public access to information and services**
  - **require timely and predictable service behaviour**
- **Increasing use of multi-media technology**
  - **require timely synchronisation between different media**

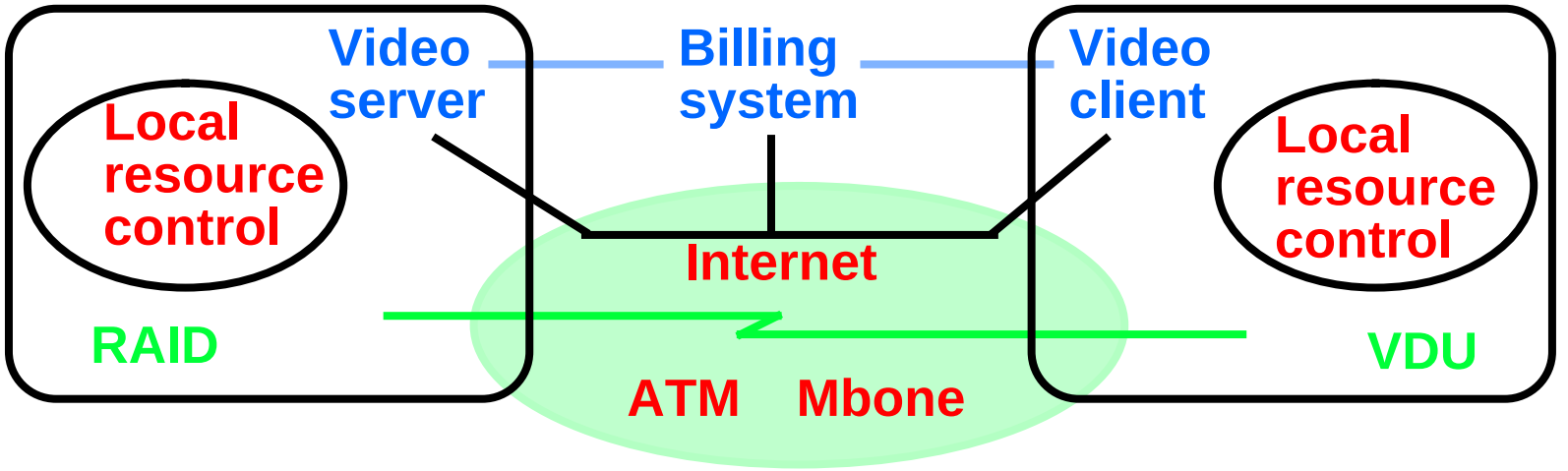
Demand is there, software is the bottleneck

Technology enablers

- Multi-media PC's are commonplace
  - multi-media GUI islands
- Network bandwidth available: ATM (B-ISDN), Mbone
  - no or little network resource management
  - no end-to-end resource management
- Affordable hardware will lead to very large scale deployment
  - multi-media experiments tend to be small scale

Technology available, but software is the bottleneck

Technical challenge

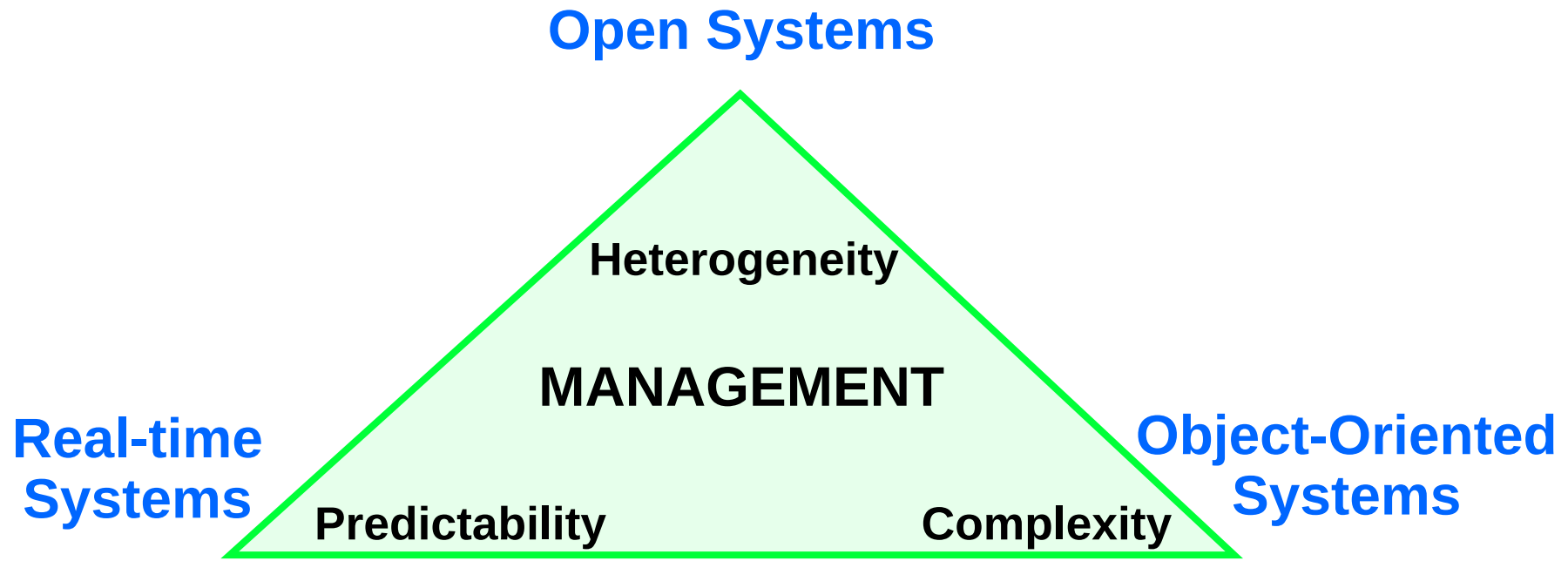


build a multi-media architecture

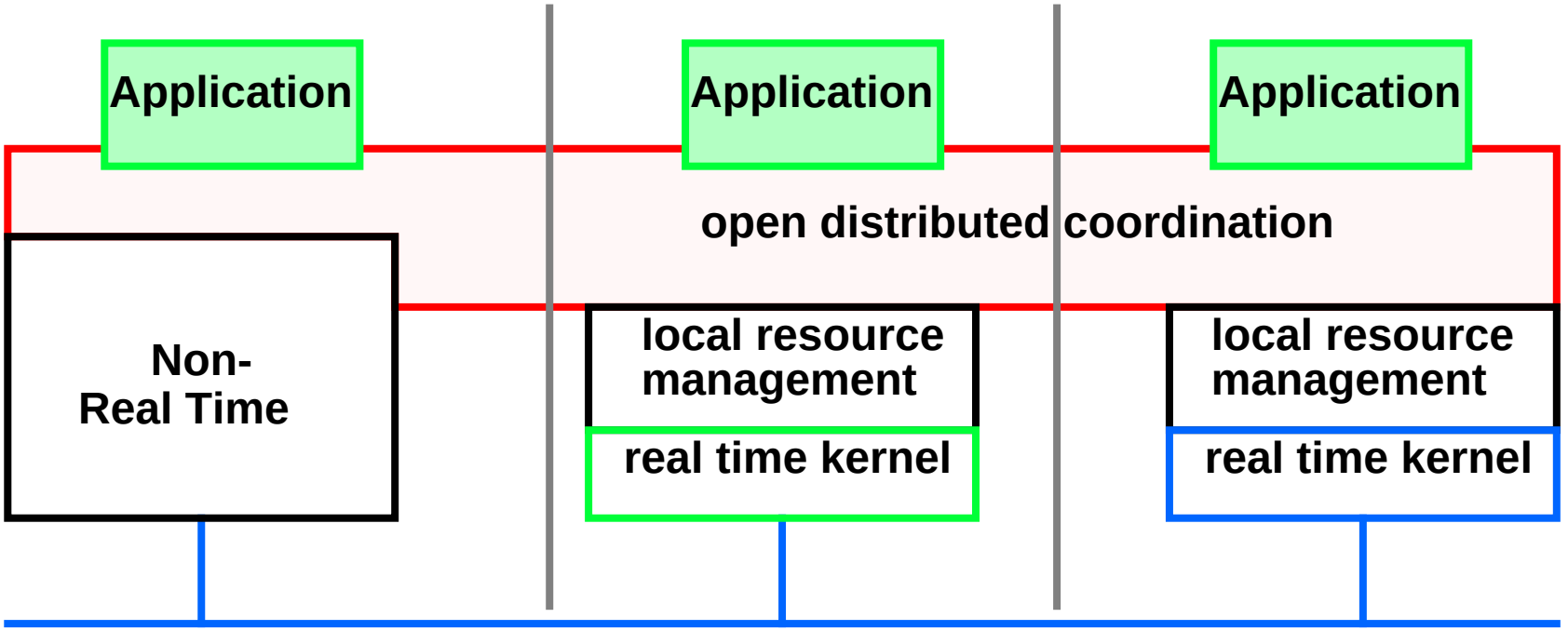
INTEGRATED

with the general purpose distributed computing architecture

First step: ANSAware/RT 1.0



ANSAware/RT 1.0



ANSAware/RT 1.0 objectives

- Application controlled resource allocation (processor and network)
- Allow interworking between applications
  - with different real-time requirements
  - on different real-time platforms
  - on real-time and non-real-time platforms
- Compatible with ANSAware 4.1
- Portability
  - existing ANSA services work on ANSAware/RT 1.0
- Use de-facto industry standard technologies
  - real-time POSIX thread package (POSIX 1003.4a)
  - p-thread scheduling and threading capabilities

Design principles

- **Resource separation**
  - resource pools (processing, memory, communications)
  - multiple scheduling policies

- **Controlled sharing**
  - map activities to resource pools
  - prioritise activities
  - set deadlines for invocations

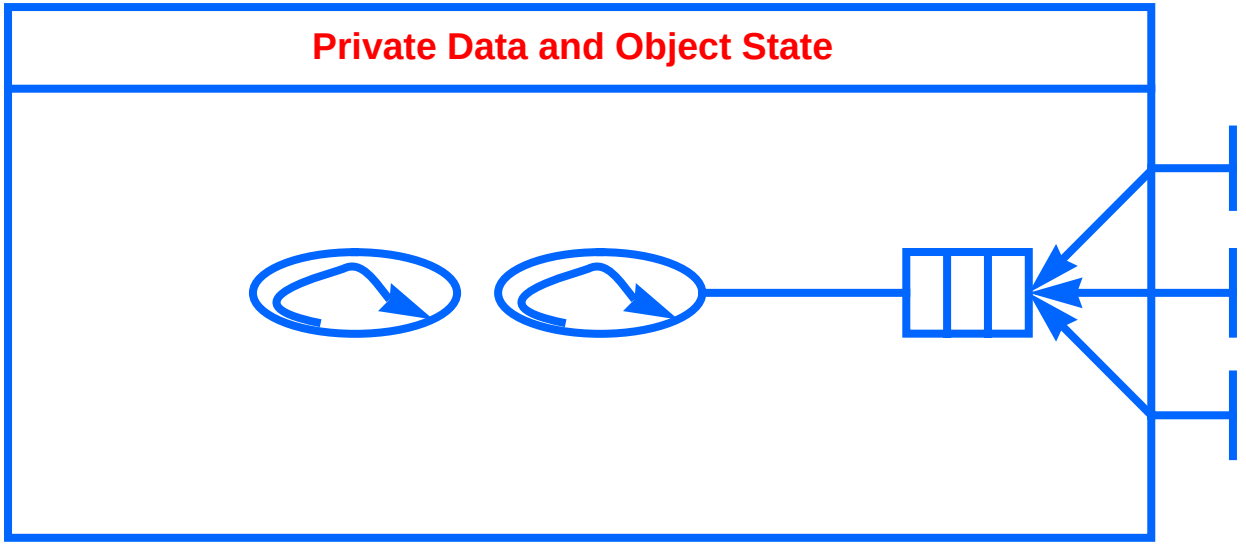
Programming Model

- **ANSAware 4.1 programming model with extensions:**
  - **Entry**
  - **Rendezvous**
  - **Explicit binding**
  - **Invocations with QoS**

best explained in combination with

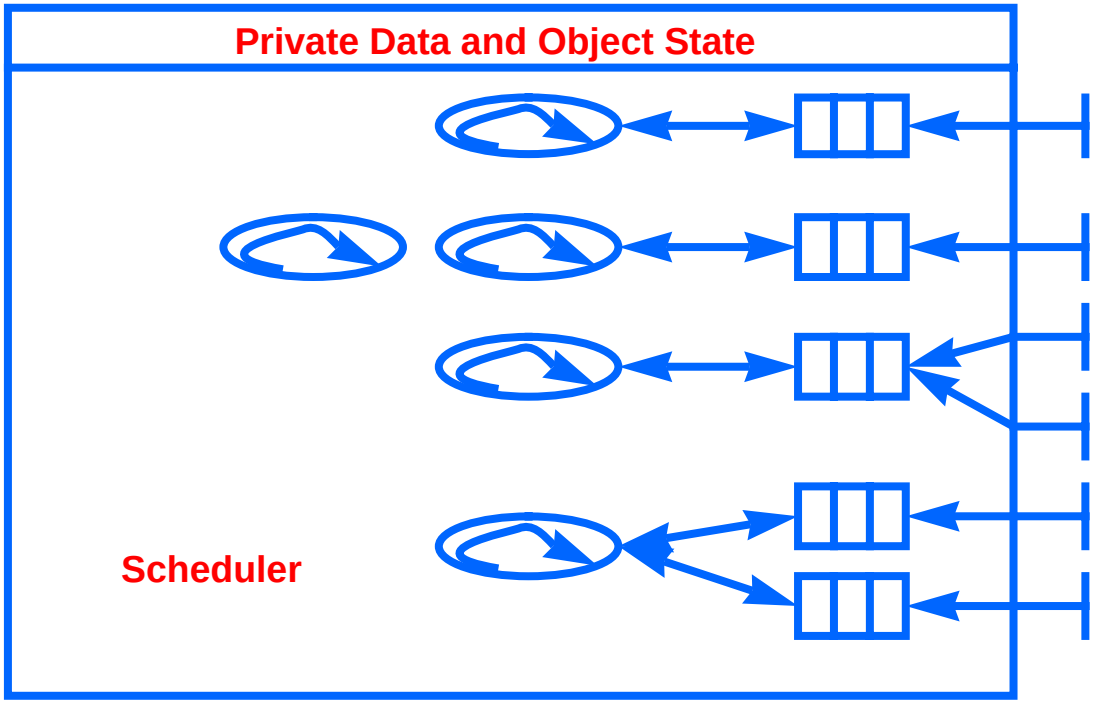
the Engineering Model

Tasking: ANSAware 4.1 Engineering Model



time-sharing based design --- efficient in resource sharing

Tasking: ANSAware/RT 1.0



more flexible design --- resource separation and reservation

Legend

Task Group

Task

Invocation
enqueue

Entry

Rendezvous

Interface

Tasking: Entry

- An Entry is a thread queue
  - a scheduling point with its own scheduling and processing policies
- Capsule has default entry to which all new interfaces are bound
- Application can create and destroy *entries*:

```
ansa_Status ansa_entry_create (ansa_Entry *entry,
                               ansa_EntryAttr attr)
```

- Bind interfaces to entries by:

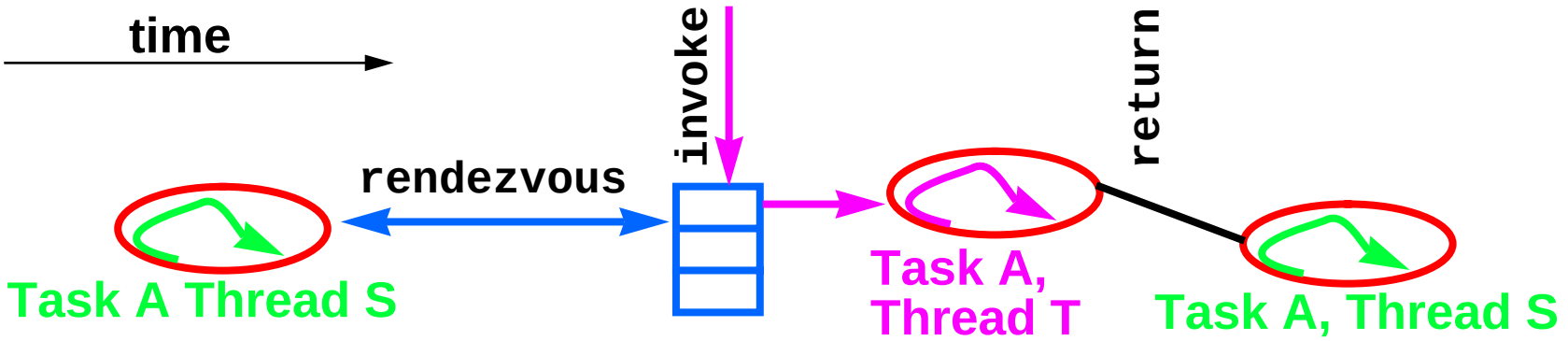
```
ansa_Status ansa_entry_bind (ansa_InterfaceRef *ref,
                             ansa_Entry *entry)
```

- Unbind to revert back to default entry

Tasking: Rendezvous

- A thread may wait at an *entry* to rendezvous and allow execution of another thread (e.g. an invocation)

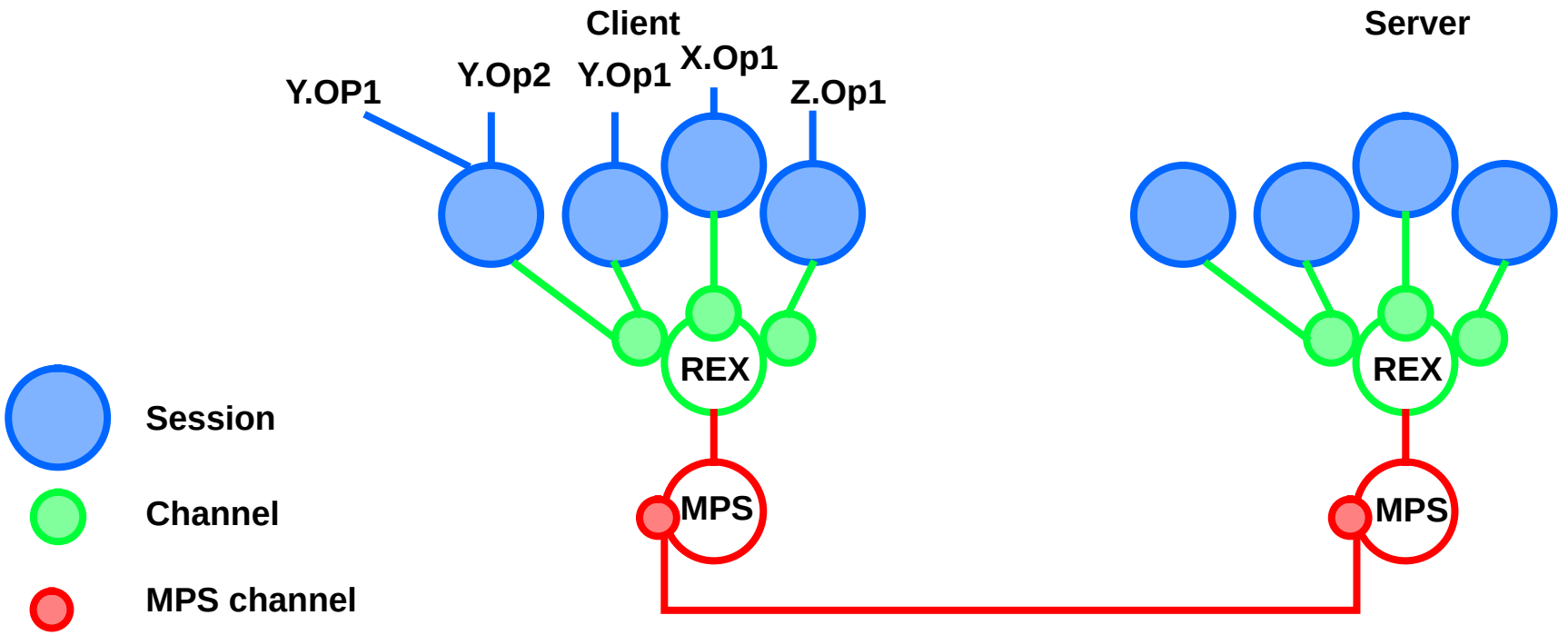
```
ansa_Status ansa_rendezvous (ansa_Entry *entry,
                             ansa_Cardinal timeout)
```



Tasking: Other functions

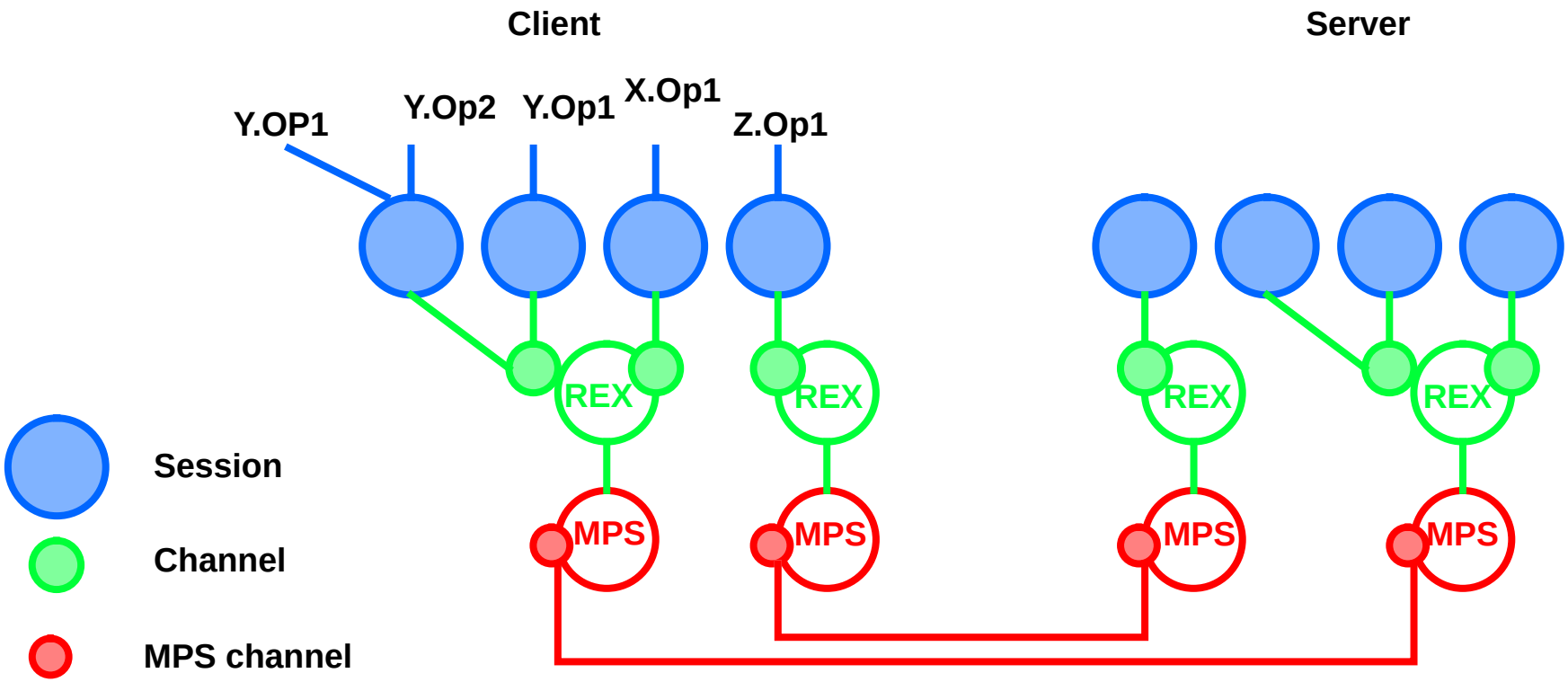
- **Application control over scheduling policy**
  - **7 real-time functions (e.g. choose task priority, select thread queuing, control task allocation to entry) for run-time control**
  - **static control via environment variable**
- **Attribute objects**
  - **describe attributes of tasks, threads or entries**
  - **similar to type**

ANSAware 4.1 Communication System



multiplexing whenever possible --- efficient resource usage

ANSAware/RT 1.0 Parallel Protocol Stacks



resource allocation to specific channels

Quality of Service (QoS)

- **QoS objects**
  - **hold attributes which describe communications resource and performance constraints**
- **End-point QoS object**
  - **used to specify the QoS characteristics of an endpoint at bind time**
- **In-band QoS object**
  - **used to specify the QoS characteristics of specific invocations at run time**

Explicit binding (1)

- Associate QoS with communications endpoints
- Control time of binding

- Server side:
- Create a service instance and set up the comms endpoint:

```
(ir) :: Type$Create(concurrency) (QoS)
```

- Destroy a service instance:

```
() :: Type$Destroy(ir)
```

Explicit binding (2)

- **Client side:**
- **Create client communications endpoint with required QoS**

```
() :: ir$Bind() (QoS)
```

- **Unbind from client communications endpoint:**

```
ir$Discard
```

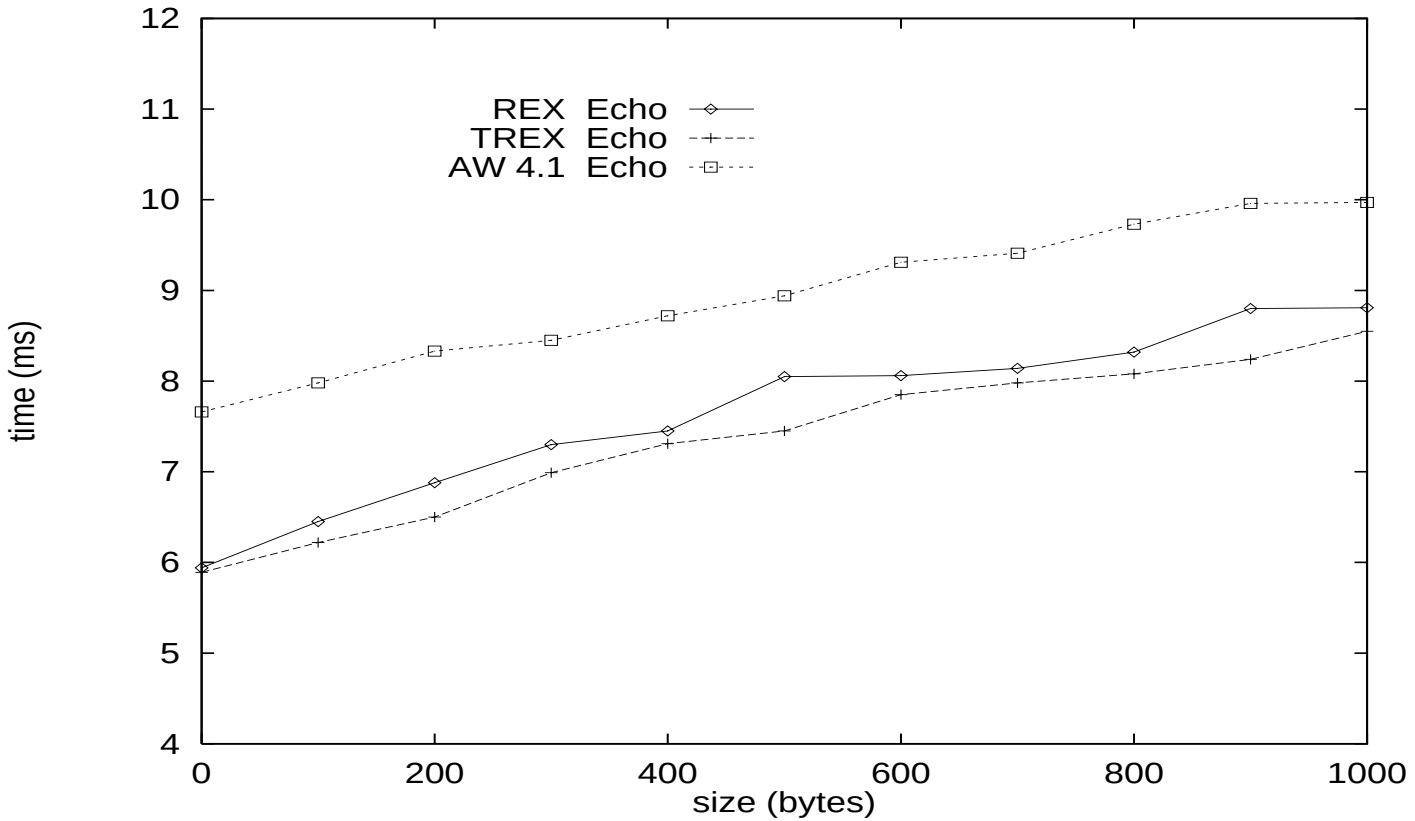
Invocation with QoS

- **Each invocation can be associated with an in-band QoS object**
  - **used to control semantics of the communication**

```
(results) <- ir$operation(arguments) signals (QoS)
```

- **Timed RPC: TREX**
  - **efficient cut-down version of REX**
  - **understands priority and deadline**

Performance



Availability

- If you require support for distributed real time objects use:

  ANSAware/RT 1.0

  over DEC Alpha OSF/1 (1.3 and 3.0)

- Windows NT version under development

- On non-real time operating systems continue to use:

  ANSAware 4.1

  (over SUNOS, HPUX, PC/DOS, Windows)

- Plan to integrate 4.1, /RT & multi-media into ANSAware 5.0 (1996)